
The Scientific Data Exchange Reference Guide

Includes implementation for x-ray tomography, x-ray fluorescence
microscopy, and x-ray photon correlation

<http://www.aps.anl.gov/DataExchange>

Version 0.0.13

April 10, 2012

Table 1: Version history

Version Date		Notes
v000	Nov 15, 2011	FdC: First version of the Data Exchange file format for full field x-ray imaging and tomography based on the definition from https://confluence.aps.anl.gov/ .
v001	Dec 23, 2011	FdC: Added sample and instrument class to meet APS (2-BM, 13-BM, 32-ID) and SLS (Tomcat) meta data requirements and definitions.
v002	Jan 5, 2012	FdC: Merged with the Coherent X-ray Imaging Data Bank file format CXI from http://cxidb.org/cxi.html . Converted the document using the same diagram definition set by Filipe Maia in "CXI file format" and modified to fit the Data Exchange definitions.
v003	Jan 15, 2012	NS: Added Provenance Class
v004	Feb 6, 2012	FdC: Added measurement class and moved sample and instrument under it to meet ESRF request to allow for multiple tomography measurements to be stored in the same file (relevant for nano CT raster scans and similar).
v005	Feb 19, 2012	FdC: Clean up 2FXi vs Data Exchange and moved to version control.
v006	Feb 21, 2012	FdC/FM (SLS): explained more clearly that the 3D array dimension order (rotation, ccd y, ccd x) is a default but not mandatory. Affected sections 3.4 , 6 , 4.4.1 , 6.2.1 .
v007	Feb 22, 2012	FdC/YP: added more info in the reconstruction and algorithm classes. Affected section 6.2.3 and table 36 .
v008	Feb 25, 2012	FdC: Expanded Data Exchange and Detector definition to add fluorescence, photon correlation. Affected section 2.3 , 6 , 8 , 7 , 6.1.1 , 4.3.2.5 , 7.1.1 .
v009	Feb 26, 2012	NS: Expanded Data Exchange and Detector definition for XPCS. Affected section 7 , 7.1.1 , 7.2.1 .

Table 1: Version history

Version Date		Notes
v010	Mar 1, 2012	NS: Expanded Data Sample, Source, and Shutter definitions as well as Detector and other definitions for XPCS. Affected section 4.3.1 , 4.3.2.2 , 4.3.2.1 , 7 , 7.1.1 , 7.2.1 , 7.1.3
v011	Mar 2, 2012	FdC: Split into technique specific the "instrument" definitions. Affected section 6.1 , 7.1 , 8.1
v012	Mar 5, 2012	NS: Added sample thickness. Expanded XPCS definition. Affected section 4.3.1 , 7.2.1
v013	Apr 5, 2012	CS: Restructuring document into sections included into main document(s). Other edits.

Contents

1	Introduction	1
2	The Design of Data Exchange	2
2.1	HDF5	3
2.2	Data types	3
2.3	Root Level Structure	3
3	Data Exchange by Example	5
3.1	Diagram color code	5
3.2	Data Exchange for Full Field X-ray Imaging	5
3.3	A minimal Data Exchange file for Imaging	5
3.4	A minimal Data Exchange file for tomography	6
3.5	A typical Data Exchange file for tomography	6
3.5.1	Sample Temperature Scan	7
3.5.2	X-ray Energy Scan	10
3.5.3	Detector-sample Distance Scan	11
3.5.4	Series of Tomographic Measurements	11
4	Data Exchange Core Reference	13
4.1	Top level (root)	13
4.2	Exchange Group	14
4.2.1	3D array data attributes	14
4.3	Measurement	15
4.3.1	Sample	15
4.3.1.1	Geometry	17
4.3.1.1.1	Translation	17
4.3.1.1.2	Orientation	18
4.3.1.2	Experiment identifier	19
4.3.1.3	Experimenter identifier	20
4.3.2	Instrument	21
4.3.2.1	Source	22
4.3.2.2	Shutter	23
4.3.2.3	Attenuator	23
4.3.2.4	Monochromator	23
4.3.2.5	Detector	25
4.4	Provenance	26
4.4.1	Process	27
4.4.2	Process description	28

5	Code Examples	29
5.1	Creating a minimal Data Exchange file	29
6	Data Exchange for X-ray Tomography	30
6.1	Instrument specific for x-ray tomography	31
6.1.1	Detector	31
6.1.2	ROI	35
6.1.3	Objective	36
6.1.4	Scintillator	37
6.1.5	Setup	38
6.1.6	Rotation Setup	39
6.1.7	Acquisition	40
6.1.8	Dark Setup	41
6.1.9	White Setup	42
6.1.10	Interferometer	43
6.2	APS 2-BM Process descriptions	43
6.2.1	Sinogram	44
6.2.2	Ring Removal	45
6.2.3	Reconstruction	46
6.2.4	Algorithm	47
6.2.5	Gridftp	49
6.2.6	Export	50
6.3	Creating a minimal Data Exchange file for tomography . .	50
6.4	Creating a typical Data Exchange file for tomography . .	50
7	Data Exchange for X-ray Photon Correlation	51
7.1	Instrument specific for x-ray photon correlation	51
7.1.1	Detector	51
7.1.2	Kinetics	54
7.1.3	Setup	55
7.2	APS Sector 8 Process descriptions	56
7.2.1	XPCS	57
8	Data Exchange for X-ray Fluorescence	60
8.1	Instrument specific for x-ray fluorescence	60
A	Appendix	61
A.1	Default units for Data Exchange entries	61
A.1.1	Angles	61
A.1.2	Dates	61

A.2 Geometry	63
A.2.1 Coordinate System	63
A.2.2 The local coordinate system of objects	63

1 Introduction

This document is a complete reference to the Data Exchange file format, including documented implementations for various beamline techniques. Briefly, Data Exchange is a set of guidelines for storing scientific data and metadata in a Hierarchical Data Format 5 (HDF5) file (<http://www.hdfgroup.org/HDF5>).

This reference guide describes the basic design principles, examples of their application, a core reference for guidelines common to most uses, and coding examples. The guide ends with a section devoted to each known implementation for a particular beamline technique.

2 The Design of Data Exchange

For various reasons, many x-ray techniques developed at synchrotron facilities around the world are unable to directly compare results due to their inability to exchange data and software tools. The aim of Data Exchange is to define a simple file format offering few basic rules and allowing each community to extend and add technique specific components. The goal is to provide simplicity and extensibility in defining data, meta data and provenance information in a simple way that can be easily adopted by various x-ray techniques.

The Data Exchange format is implemented using Hierarchical Data Format 5 (HDF5), which offers platform-independent binary data storage with optional compression, hierarchical data ordering, and self-describing tags so that one can examine a HDF5 file's contents with no knowledge of how the file writing program was coded.

The aim and the scope of Data Exchange is very similar to the Coherent X-ray Imaging Data Bank file format (CXI), so whenever possible we will use the same conventions, name tags, and reference system. This document is using a similar diagram definition set by Filipe R. N. C. Maia in "CXI file format" (<http://cxidb.org/cxi.html>), with a few minor modifications for Data Exchange definitions.

The core principle of Data Exchange is that it must be simple enough that it is not necessary to use a support library beyond core HDF5. The simplicity of Data Exchange read and write is achieved using basic HDF5 calls, making it easy for anyone to either look at an example file using `h5dump` or `HDFView`, or to look at example code in language X, and then create their own read and write routines in language Y.

The simplest Data Exchange file provides information sufficient to share a multidimensional data array. In this simplest form, Data Exchange implements only one "exchange" group. The "exchange" definition is designed to allow for simple exchange of images, spectra, and other forms of beamline detector data with a minimum of fields. This definition is essentially a technique-agnostic format for exchanging data with others. Data Exchange is also designed to be extended to include technique-specific data and metadata information. This is achieved by providing optional, but clearly defined, metadata components to the base definition.

2.1 HDF5

The HDF5 format is the basis of the Data Exchange format. Data Exchange, like CXI, is not a completely new file format, but simply a set of rules designed to create HDF5 files with a common structure to allow a uniform and consistent interpretation of such files.

HDF5 was chosen as the basis because it is a widely used high performance scientific data format which many programs can already, at least partially, read and write. It also brings with it the almost automatic fulfillment of the Data Exchange requirements, i.e. simplicity, flexibility and extensibility. HDF5 version 1.8 or higher is required as previous versions don't support all features required by Data Exchange.

2.2 Data types

Data Exchange uses the same CXI convention for data types as defined at <http://cxidb.org/cxi.html> using HDF5 native datatypes. The data should be saved in the same format as it was created/acquired. For example CCD images acquired as 16 bit integers should be saved using the `H5T_NATIVE_SHORT` HDF5 type. In this way all cross platform big-little endian issues reading and writing files are eliminated.

2.3 Root Level Structure

While HDF5 gives great flexibility in data storage, straightforward file readability and exchange requires adhering to an agreed-upon naming and organizational convention. To achieve this goal, Data Exchange adopts a layered approach by defining a set of *mandatory* and *optional* fields.

The general structure of a Data Exchange file is shown in table 2. The most basic file must have an "implements" string, and an "exchange" group at the root level/group of the HDF5 file. Optional "measurement" and "provenance" groups are also defined. Beyond this, additional groups may be added to meet individual needs, with guidelines suggesting the best structure.

implements

Mandatory scalar string dataset in the root of the HDF5 file whose value is a colon separated list that shows which components are present in the file. All components listed in the *implements* string are to be groups placed in the HDF5 file at the root level/group.

Table 2: Data Exchange general rules

Member	Type	Example
<i>implements</i>	string dataset	"exchange:measurement:provenance"
<i>exchange</i>	group	
<i>measurement</i>	group	
<i>provenance</i>	group	

In a minimal Data Exchange file, the only mandatory item in this list is *exchange*. A more general Data Exchange file also contain *measurement* and possibly *provenance*, in which case the implements string would be: "exchange: measurement: provenance"

exchange

Mandatory group containing one or more arrays that represent the most basic version of the data, such as raw or normalized optical density maps or a elemental signal map. *Exchange_N* is used when more than one core dataset or derived datasets are saved in the file. The *exchange* implementation for specific techniques are defined in separate sections in the Reference Guide.

measurement

Optional group containing the measurement made on the sample. *Measurement* contains information about the sample and the instrument. *Measurement_N* is used when more than one measurement is stored in the same file.

provenance

Optional group containing information about the status of each processing step.

In a Data Exchange file, each dataset has a unit defined using the *units* attribute. *units* is not mandatory - if omitted, the default unit as defined in [Appendix A.1](#) is used.

The detailed rules about how to store datasets within the exchange group are best shown through examples in the next section. Detailed reference information can be found in the [Data Exchange Core Reference](#) section.

3 Data Exchange by Example

3.1 Diagram color code

The diagrams of the Data Exchange file follow the same color conventions used by the CXI and reported in Figure 1

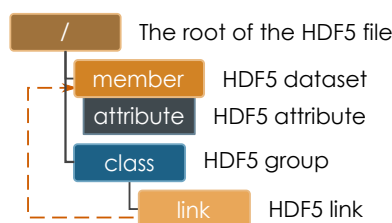


Figure 1: Explanation of the color code used in the diagrams

3.2 Data Exchange for Full Field X-ray Imaging

The data file format for full field x-ray imaging (2FXi) is defined as the Data Exchange implementation to store all experimental data collected during full field x-ray imaging and tomography experiments as well as to capture infrastructure meta data and data provenance by recording how the data were acquired, processed and transferred.

The 2FXi file format complies with the Data Exchange file format defined in Section 2.3, adding, as required by the Data Exchange definition, the technique-specific groups for full field x-ray imaging and tomography.

The goal of Data Exchange implementation for x-ray full field imaging is to provide simplicity and extensibility in defining data, meta data and provenance information for x-ray imaging, micro and nano tomography.

3.3 A minimal Data Exchange file for Imaging

Figure 2 shows a diagram a minimal Data Exchange file to store a single projection image. As no units are specified the data is assumed to be in “counts” with the axes (x, y) in pixels.

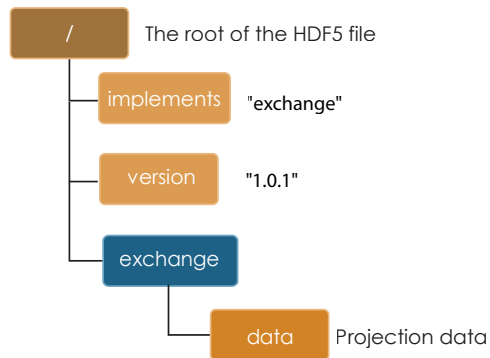


Figure 2: Diagram of a minimal Data Exchange file for a single image.

3.4 A minimal Data Exchange file for tomography

A tomographic data set consists of a series of projections, dark and white field images. The dark and white fields must have the same projection image dimensions and can be collected at any time before, after or during the projection data collection. The angular position of the tomographic rotation axis is used to keep track of when the dark and white images are collected. 2FXi saves projection images, dark and white images in three 3D arrays as shown in Figure 3 and 4 using, by default, the natural HDF5 order of the a multidimensional array (rotation axis, ccd y, ccd x), i.e. with the fastest changing dimension being the last dimension, and the slowest changing dimension being the first dimension. If using the default dimension order, the axes attribute (see Table 4) "theta:y:x" can be omitted. The `axes` attribute is mandatory if the 3D arrays use a different axes order. This could be the case when, for example, the arrays are optimized for sinogram read (`axes = "y:theta:x"`). As no units are specified the data is assumed to be in "counts" with the axes (x, y) in pixels.

3.5 A typical Data Exchange file for tomography

A series of tomographic data sets are typically collected changing the instrument status (energy, detector or optics position) or changing the sample status (position, environment etc.). Figure 5, 6 and 7 show the content of 2FXi files changing the sample temperature, the x-ray source energy and detector-sample distance.

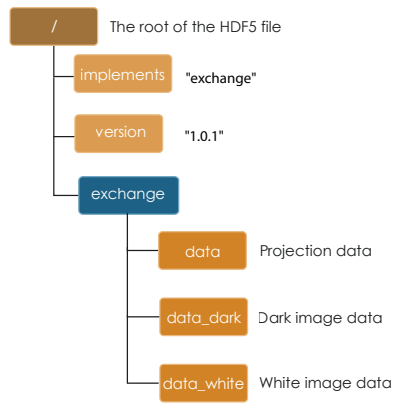


Figure 3: Diagram of a minimal Data Exchange file for a single tomographic data set including raw projections, dark and white fields. Since the positions of the rotation axis for each projection, dark and white images are not specified is assumed that the raw projections are taken at equally spaced angular intervals between 0 and 180 degree, with white and dark field collected at the same time before or after the projection data collection.

3.5.1 Sample Temperature Scan

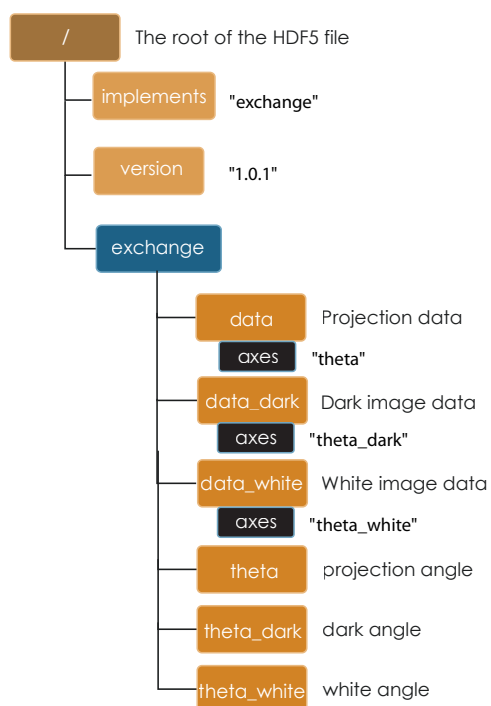


Figure 4: Diagram of a minimal Data Exchange file for a single tomographic data set including raw projections, dark and white fields. In this case the attribute `axes` indicates the presence of theta vectors containing the positions of the rotation axis for each projection, dark and white images.

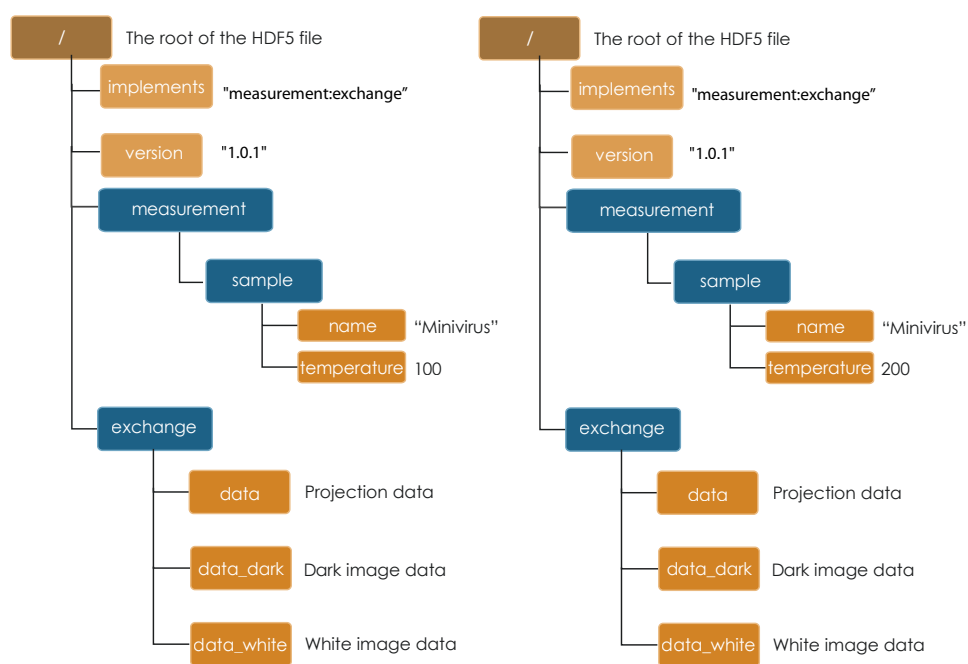


Figure 5: Diagram of two tomographic data sets taken at two different sample temperatures (100 and 200 K). To store the temperature in °C is necessary to add the attribute `units = "celsius"` to the `temperature` tag.

3.5.2 X-ray Energy Scan

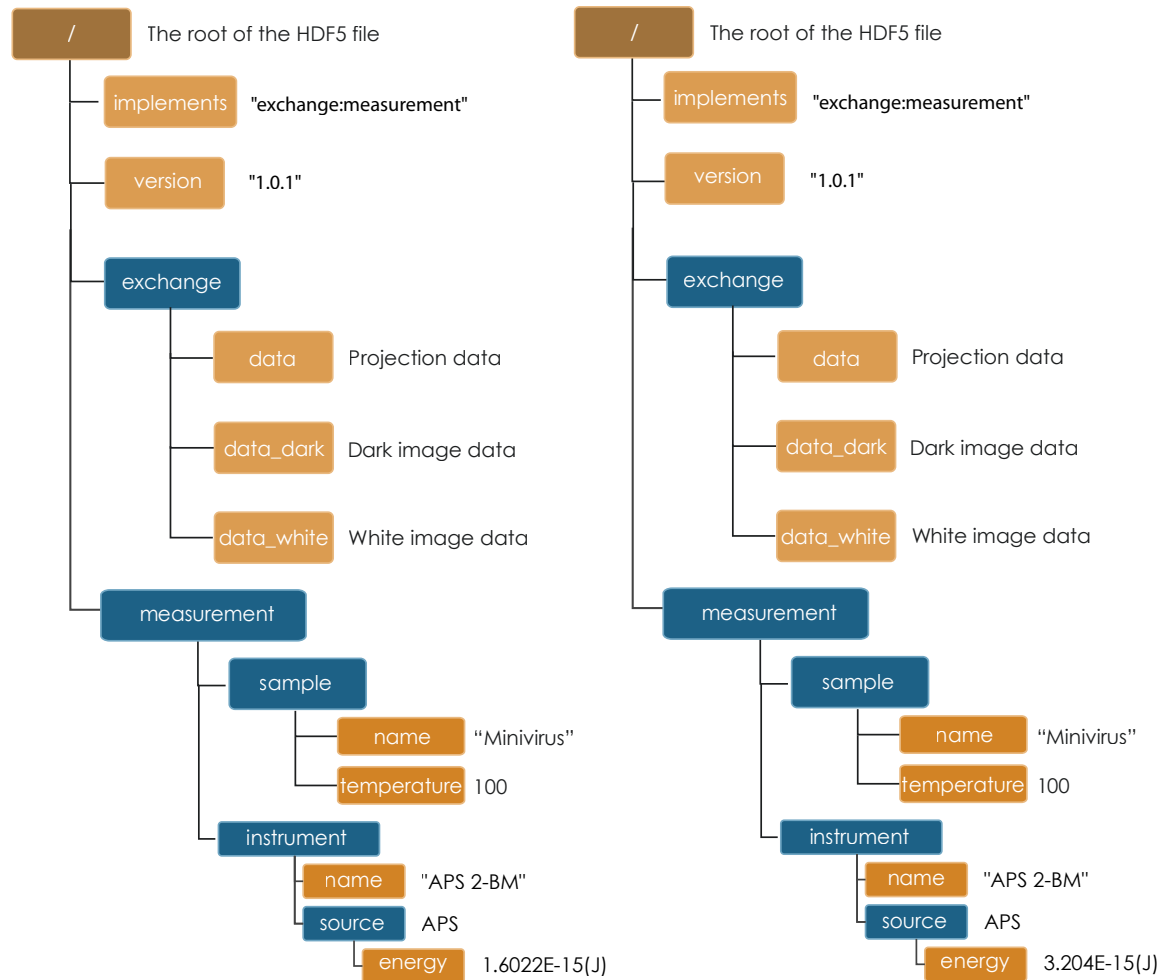


Figure 6: Diagram of two tomographic data sets taken at two different energy (10 and 20 keV). To store the temperature in *keV* is necessary to add the attribute `units = "keV"` to the energy tag.

3.5.3 Detector-sample Distance Scan

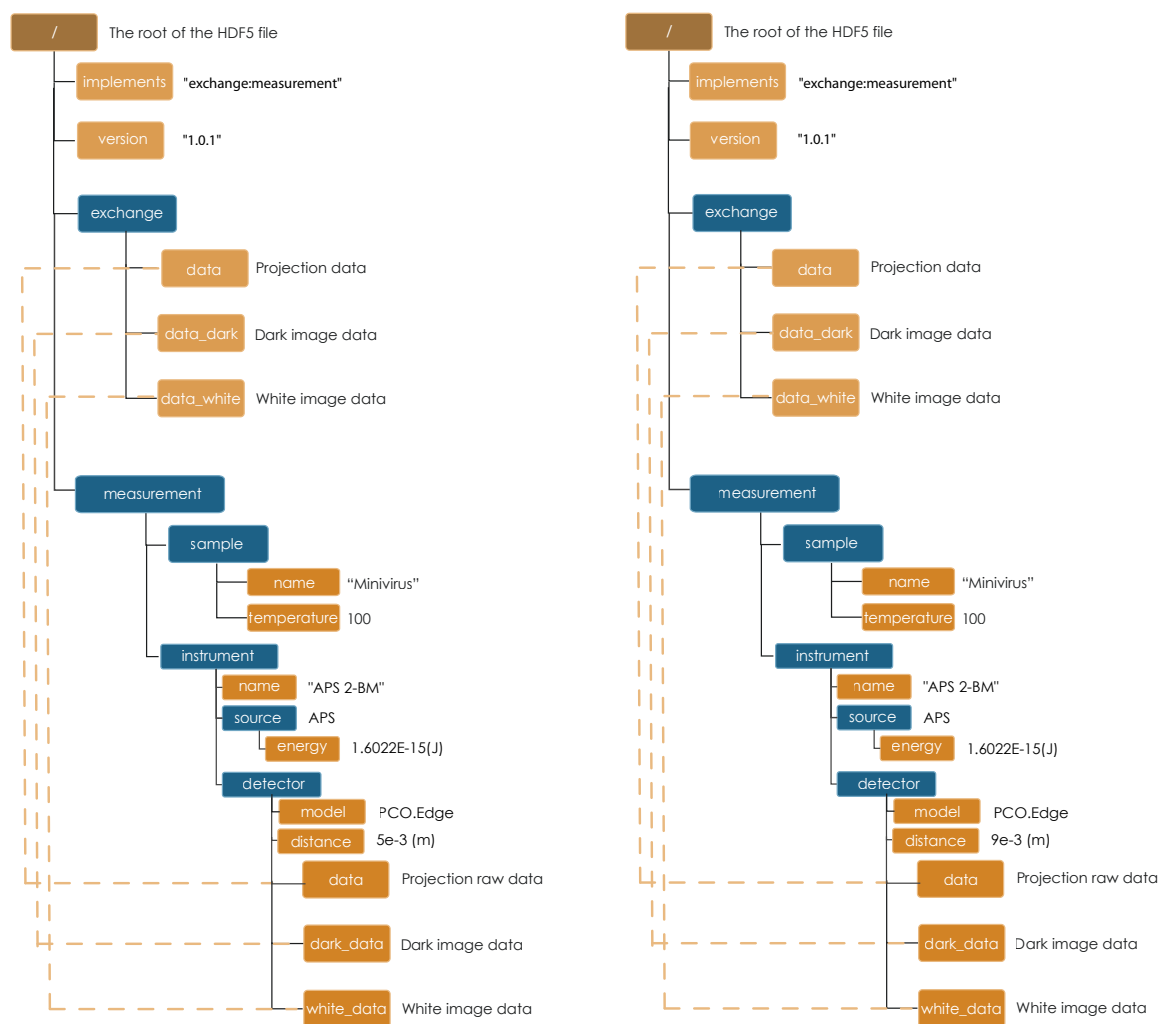


Figure 7: Diagram of two tomographic data sets collected with two different detector-sample distances (5 and 9 mm).

3.5.4 Series of Tomographic Measurements

A series of tomographic measurements, when relevant, can be stored in the same 2FXi file appending $_N$ to the measurement tag. For example in nano tomography experiments the detector field of view is often

smaller than the sample. To collect a complete tomographic data set is necessary to raster the sample across the field of view moving its x and y location. Figure 8 shows a 2FXi file from a nano tomography experiment when the sample rasters through the field of view. The details of how the *exchange* arrays for a raster nano tomography scan are generated will be discussed in more details in Section 4.4.

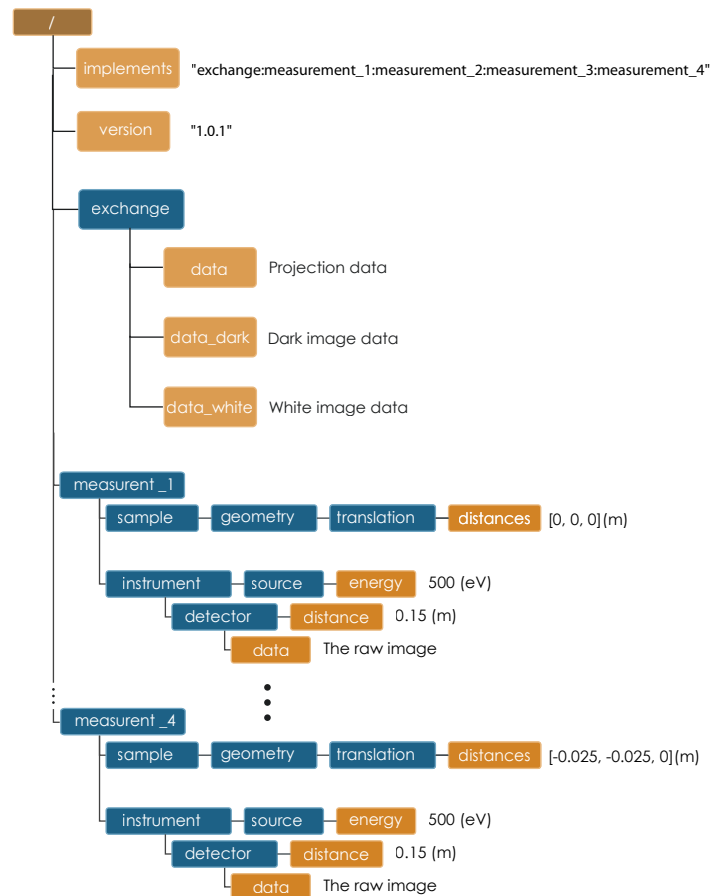


Figure 8: Diagram of a 2FXi file with 4 tomographic data sets from a nano tomography experiment.

4 Data Exchange Core Reference

4.1 Top level (root)

This node represents the top level of the HDF5 file and holds some general information about the file.

Table 3: Data Exchange top level entries

Member	Type	Example
<i>implements</i>	string	"exchange:measurement:provenance"
<i>version</i>	string	"1.0.1"
<i>exchange_N</i>	Exchange group	
<i>measurement_N</i>	Measurement group	
<i>provenance</i>	Provenance group	

implements - A colon separated list that shows which components are present in the file. The only *mandatory* component is *exchange*. A more general Data Exchange file also contains *measurement* and *provenance* information, if so these will be declared in *implements* as "exchange:measurement:provenance"

version - Data Exchange format version.

exchange_N - The data taken from measurements or processing.

measurement_N - Each measurement made on the sample.

provenance - The Provenance class describes all process steps that have been applied to the data.

4.2 Exchange Group

This class is a general placeholder for the most important information in a Data Exchange file and contains one or more arrays representing the most basic version of the data.

4.2.1 3D array data attributes

Table 4: data attributes

Member	Type	Example
description	string	"transmission"
units	string	"counts"
axes	string	"theta:y:x"

Table 5: x, y, theta, theta_dark, theta_white attribute

Member	Type	Example
units	string	" μm ", "degree"

4.3 Measurement

This class holds sample and instrument information.

Table 6: Data Exchange top level entries

Member	Type	Example
sample	Sample class	
instrument	Instrument class	

sample - The sample measured.

instrument - The instrument used to collect this data.

4.3.1 Sample

This class holds basic information about the sample, its geometry, properties, the sample owner (user) and sample proposal information.

Table 7: Sample class members

Member	Type	Example
name	string	"cells sample 1"
description	string	"malaria cells"
preparation_date	string ISO 8601	"2011 07 15T15 10Z"
chemical_formula	string abbr. CIF format	"(Cd 2+)3, 2(H2 O)"
mass	float	0.25
concentration	float	0.4
environment	string	"air"
temperature	float	25.4
temperature_set	float	26.0
pressure	float	101325
thickness	float	0.001
position	string	"2D" APS robot coord.
geometry	Geometry group	
ids	Experiment identifier group	
experimenter	Experimenter identifier group	

name - Descriptive name of the sample.

description - Description of the sample.

preparation_date - Date and time the sample was prepared.

chemical formula - Sample chemical formula using the CIF format.

mass - Mass of the sample.

concentration - Mass/volume.

environment - Sample environment.

temperature - Sample temperature.

temperature_set - Sample temperature set point.

pressure - Sample pressure.

thickness - Sample thickness.

position - Sample position in the sample changer/robot.

geometry - Sample center of mass position and orientation.

ids - Facility experiment identifiers.

experimenter - Experimenter identifiers.

4.3.1.1 Geometry This class holds the general position and orientation of a component.

Table 8: Geometry class members

Member	Type	Quantity
translation	Translation class	
orientation	Orientation class	

orientation - The rotation of the object with respect to the coordinate system.

translation - The position of the object with respect to the origin.

Only one orientation and one translation is permitted in each geometry class.

The position of the origin of the object should be explicitly defined for each object. If it is not defined it should be assumed to be the center of the object.

4.3.1.1.1 Translation This is the description for the general spatial location of a component - it is used by the Geometry class

Table 9: Translation class members

Member	Type	Example
distances	3 floats	0, 0.001, 0

distances - The x, y and z components of the translation of the origin of the object relative to the origin of the global coordinate system (the place where the X-ray beam meets the sample when the sample is first aligned in the beam). If distances does not have the attribute units set then the units are in meters (see table 45)

4.3.1.1.2 Orientation This is the description for a general orientation of a component - it is used by the Geometry class.

Table 10: Orientation class members

Member	Type	Quantity
value	6 floats	unitless

value - Dot products between the local and the global unit vectors.

The orientation information is stored as direction cosines. The direction cosines will be between the local coordinate directions and the global coordinate directions. The unit vectors in both the local and global coordinates are right-handed and orthonormal.

Calling the local unit vectors (x', y', z') and the reference unit vectors (x, y, z) the six numbers will be $[x' \cdot x, x' \cdot y, x' \cdot z, y' \cdot x, y' \cdot y, y' \cdot z]$ where “ \cdot ” is the scalar dot product (cosine of the angle between the unit vectors).

Notice that this corresponds to the first two rows of the rotation matrix that transforms from the global orientation to the local orientation. The third row can be recovered by using the fact that the basis vectors are orthonormal.

Table 11: Experiment identifier class

Member	Type	Example
proposal	string	"1234"
activity	string	"9876"
safety	string	"9876"

4.3.1.2 Experiment identifier **proposal** - Proposal reference number. For the APS this is the General User Proposal number.

activity - Proposal scheduler id. For the APS this is the beamline scheduler activity id.

safety - Safety reference document. For the APS this is the Experiment Safety Approval Form number.

Table 12: Experimenter identifier class

Member	Type	Example
name	string	"John Doe"
role	string	"Project PI"
affiliation	string	"University of California, Berkeley"
address	string	"EPS UC Berkeley CA 94720 4767 USA"
phone	string	" +1 123 456 0000"
email	string	"johndoe@berkeley.edu"
facility_user_id	string	"a123456"

4.3.1.3 Experimenter identifier **name** - User name.

role - User role.

affiliation - User affiliation.

address - User address.

phone - User phone number.

email - User e-mail address

facility_user_id - User badge number

4.3.2 Instrument

The instrument group stores all relevant beamline components status at the beginning of the tomographic measurement.

Table 13: Instrument group members

Member	Type	Example
name	string	"XSD/2-BM"
source	Source group	
shutter_ <i>N</i>	Shutter group	
attenuator_ <i>N</i>	Attenuator group	
monochromator	Monochromator group	
interferometer	Interferometer group	
detector_ <i>N</i>	Detector group	
sample_stack	Sample Stack group	
acquisition	Acquisition group	

name - Name of the instrument.

source - The source used by the instrument.

shutter_*N* - The shutter(s) used by the instrument.

attenuator_*N* - The attenuators that are part of the instrument.

monochromator - The monochromator used by the instrument.

detector_*N* - The detectors that compose the instrument.

4.3.2.1 Source Class describing the light source being used.

Table 14: Source class members

Member	Type	Example
name	string	"APS"
datetime		
beamline	string	"2-BM"
distance	float	-48.5
current	float	0.094
energy	float	4.807e-15
pulse_energy	float	1.602e-15
pulse_width	float	15e-11
mode	string	"TOPUP"
beam_intensity_incident	float	55.93
beam_intensity_transmitted	float	100.0

name - Name of the facility.

datetime - Date and time source was measured.

beamline - Name of the beamline.

distance - The source distance (m) from the sample.

current - Electron beam current (A).

energy - Characteristic photon energy of the source (J). For an APS bending magnet this is 30 keV or 4.807e-15 J.

pulse_energy - Sum of the energy of all the photons in the pulse (J).

pulse_width - Duration of the pulse (s).

source - Beam mode: TOPUP.

beam_intensity_incident - Incident beam intensity in (photons per s).

beam_intensity_transmitted - Transmitted beam intensity (photons per s).

4.3.2.2 Shutter

Class describing the light source being used.

Table 15: Source class members

Member	Type	Example
name	string	"Front End Shutter 1"
distance	float	-48.5
status	string	"OPEN"

name - Shutter name.

distance - Shutter distance (m) from the sample.

status - "OPEN" or "CLOSED" or "NORMAL"

4.3.2.3 Attenuator

This class describes a beamline attenuator(s) used during data collection. If more than one attenuators are used they will be named as attenuator_1, attenuator_2 etc.

Table 16: Attenuator class members

Member	Type	Example
distance	float	-35.7
thickness	float	1e-3
attenuator_transmission	float	unit-less
type	string	Al

distance - The Attenuator distance (m) from the sample. Negative distances represent beamline components that are before the sample while positive distances represent components that are after the sample. In this case the filter is located 35.7 m upstream of the sample.

thickness - Thickness of attenuator along beam direction.

attenuator_transmission - The nominal amount of the beam that gets through (transmitted intensity)/(incident intensity).

type - Type or composition of attenuator.

4.3.2.4 Monochromator

Define a monochromator used in the instrument.

type - Multilayer type.

Table 17: Monochromator class members

Member	Type	Example
type	string	"Multilayer"
energy	float	1.602e-15
energy_error	float	1.602e-17
mono_stripe	string	"Ru/C"

energy - Peak of the spectrum that the monochromator selects. Since units is not defined this field is in J and corresponds to 10 keV.

energy_error - Standard deviation of the spectrum that the monochromator selects. Since units is not defined this field is in J.

mono_stripe - Type of multilayer coating or crystal.

4.3.2.5 Detector This class holds information about the detector used during the experiment. If more than one detector are used they will be all listed as detector_*N*. In x-ray fluorescence

Table 18: X-ray fluorescence detector class members

Member	Type	Example
manufacturer	string	"Cooke Corporation"
model	string	"pco dimax"
serial_number	string	"1234XW2"

manufacturer - The detector manufacturer.

model - The detector model.

serial_number - The detector serial number .

4.4 Provenance

The documentation of all transformations, analyses and interpretations of data is called data provenance. Maintaining this history allows for reproducible data. The Data Exchange format tracks provenance using the Provenance class. The index value attached to the process class denotes the execution order of the processes. The Process class uses references to other classes that describe the analysis in detail. The Provenance class describes all process steps that have been applied to the data.

Table 19: Provenance class members

Member	Type	Example
<code>process_N</code>	Process class	

`process_N` - A process applied to the data.

4.4.1 Process

The process class holds basic information about a process. It is a generic container for recording the status of a process, and maintaining references to detailed process information.

Table 20: Process class members

Member	Type	Example
status	string	"SUCCESS"
reference	string	"/reconstruction"
message	string	"Full reconstruction."

status - Current process status. May be one of the following: QUEUED, RUNNING, FAILED, or SUCCESS.

reference - Path to a process description group. The process description group contains all metadata to perform or run the specific process.

message - A process specific message generated by the process. It may be a confirmation that the process was successful, or a detailed error message, for example.

Table 21: Process class examples

process_1		
status	"SUCCESS"	
reference	"/sinogram"	
message	"modify axes from "theta:y:x" to "y:theta:x"	
process_2		
status	"SUCCESS"	
reference	"/ring_removal"	
message	"Ring removal algorithm."	
process_3		
status	"SUCCESS"	
reference	"/reconstruction"	
message	"Full reconstruction."	
process_4		
status	"RUNNING"	
reference	"/gridftp"	
message	"cluster to remote user data transfer"	
process_5		
status	"RUNNING"	
reference	"/export"	
message	"reconstructed data conversion"	

4.4.2 Process description

The process description group defined in the reference tag in each process steps contains all parameters, including input and output datasets to execute a specific process steps and it should be placed at the root of the HDF5 file.

5 Code Examples

All the code examples as well as the resulting Data Exchange files are available from <http://www.aps.anl.gov/DataExchange/>.

5.1 Creating a minimal Data Exchange file

Include code here

The resulting file should be equivalent to the one in Fig. 2.

6 Data Exchange for X-ray Tomography

In -ray tomography the 3D arrays representing the most basic version of the data include projections, dark and white fields. It is *mandatory* that there is at least one data class in each exchange class. Most data analysis and plotting programs will primarily focus in this class.

Table 22: exchange class members for tomography data

Member	Type	Example
title	string	"raw absorption tomo"
data	3D array	see 4 for attributes
x	vector of dims 2	see 5 for attributes)
y	vector of dims 1	see 5 for attributes)
theta	vector of dims 0 of data	see 5 for attributes)
data_dark	3D array	see 4 for attributes
theta_dark	vector for dims 0 of data_dark	see 5 for attributes)
data_white	3D array	see 4 for attributes
theta_white	vector for dims 0 of data_white	see 5 for attributes)

title - This is the data title.

data - A tomographic data set consists of projection, dark and white images. *Data* is a three-dimensional array containing the raw projection images. All multidimensional arrays are stored by default using the default HDF5 axes order, i.e. with the fastest changing dimension being the last dimension, and the slowest changing dimension being the first dimension. 2FXi follows this convention saving the projection data as a 3D array with dimension order: rotation, ccd y, ccd x. If using the default dimension order, the axes attribute "theta:y:x" (see Table 4) can be omitted. The `axes` attribute is mandatory if the 3D arrays use a different axes order. This could be the case when, for example, the arrays are optimized for sinogram read (`axes = "y:theta:x"`). As no units are specified the data is assumed to be in "counts" with the axes (x, y) in pixels. *Data* attributes, if used, are defined in Table 4, if *data* does not have any attributes defined then the unit is assumed to be in "counts", the axes (x, y) are in pixels.

data_dark, data_white - The dark field and white fields must have the same dimensions of the projection images and can be collected at any time before, after or during the projection data collection. *Data_dark* and *data_white* attributes, if used, are defined in Table 4, if *data_dark* and *data_white* don't have any attributes defined then the correspond-

ing `data_dark` and `data_white` are assumed to be collected all at the beginning or at the end of the projection data collection.

`x`, `y` - `x` and `y` are vectors storing the dimension scale for the second and third data array dimension. If `x`, `y` are not defined the second and third dimensions of the data array are assumed to be in pixels.

`theta`, `theta_dark`, `theta_white` - `theta` is a vector storing the projection angular positions with attributes defined in Table 5. If `theta` is not defined the projections are assumed to be collected at equally spaced angular interval between 0 and 180 degree. The dark field and white fields can be collected at any time before, after or during the projection data. `theta_dark`, and `theta_white`, with attributes defined in Table 5, store the position of the tomographic rotation axis when the corresponding dark and white images are collected. If `theta_dark` and `theta_white` are missing the corresponding `data_dark` and `data_white` are assumed to be collected all at the beginning or at the end of the projection data collection.

6.1 Instrument specific for x-ray tomography

6.1.1 Detector

This class holds information about the detector used during the experiment. If more than one detector are used they will be all listed as `detector_N`. In full field imaging the detector consists of a CCD camera, microscope objective and a scintillator screen. Raw data recorded by a detector as well as its position and geometry should be stored in this class.

`manufacturer` - The detector manufacturer.

`model` - The detector model.

`serial_number` - The detector serial number .

`bit_depth` - The detector bit depth.

`x_pixel_size`, `y_pixel_size` - Physical detector pixel size (m).

`x_dimension`, `y_dimension` - The detector horiz./vertical dimension.

`x_binning`, `y_binning` - If the data are collected binning the detector `x_binning` and `y_binning` store the binning factor.

`operating_temperature` - The detector operating temperature (K).

Table 23: X-ray tomography detector class members

Member	Type	Example
manufacturer	string	"Cooke Corporation"
model	string	"pco dimax"
serial_number	string	"1234XW2"
bit_depth	integer	12
x_pixel_size	float	6.7e-6
y_pixel_size	float	6.7e-6
x_dimension	integer	2048
y_dimension	integer	2048
x_binning	integer	1
y_binning	integer	1
operating_temperature	float	270
exposure_time	float	1.7e-3
frame_rate	integer	2
distance	float	5.7e-3
data	3D array	variable (see Tab. 4 for attrib.)
x	vector of dims 2	variable (see Tab. 5 for attrib.)
y	vector of dims 1	variable (see Tab. 5 for attrib.)
theta	vector of dims 0 of data	variable (see Tab. 5 for attrib.)
data_dark	3D array	variable (see Tab. 4 for attrib.)
theta_dark	vector for dims 0 of data_dark	variable (see Tab. 5 for attrib.)
data_white	3D array	variable (see Tab. 4 for attrib.)
theta_white	vector for dims 0 of data_white	variable (see Tab. 5 for attrib.)
roi	roi class	
objective_N	objective class	
scintillator	scintillator class	
counts_per_joule	float	unitless
basis_vectors	float array	length
corner_position	3 floats	length

exposure_time - The detector exposure time (s).

frame_rate - The detector frame rate (fps). This parameter is set for fly scan

distance - The detector distance from the sample.

data - A tomographic data set consists of projection, dark and white images. *Data* is a three-dimensional array containing the raw projection images. As defined in the HDF5 standard, all multidimensional arrays must be stored with the fastest changing dimension being the last dimension, and the slowest changing dimension being the first dimension. 2FXi follows this convention saving the projection data as a 3D array with dimension order: rotation, ccd y, ccd x. *Data* attributes, if used,

are defined in Table 4, if *data* does not have any attributes defined then the unit is assumed to be in “counts”, the axes (x, y) are in pixels and the projections are assumed to be collected at equally spaced angular interval between 0 and 180 degree.

data_dark, data_white - The dark field and white fields must have the same dimensions of the projection images and can be collected at any time before, after or during the projection data collection. Data_dark and data_white attributes, if used, are defined in Table 4, if data_dark and data_white don't have any attributes defined then the corresponding data_dark and data_white are assumed to be collected all at the beginning or at the end of the projection data collection.

x, y - X and y are vectors storing the dimension scale for the second and third data array dimension. If x, y are not defined the second and third dimensions of the data array are assumed to be in pixels.

theta, theta_dark, theta_white - Theta is a vector storing the projection angular positions with attributes defined in Table 5. If theta is not defined the projections are assumed to be collected at equally spaced angular interval between 0 and 180 degree. The dark field and white fields can be collected at any time before, after or during the projection data. Theta_dark, and theta_white, with attributes defined in Table 5, store the position of the tomographic rotation axis when the corresponding dark and white images are collected. If theta_dark and theta_white are missing the corresponding data_dark and data_white are assumed to be collected all at the beginning or at the end of the projection data collection.

roi - The detector selected Region Of Interest (ROI).

objective_N - List of the visible light objectives mounted between the detector and the scintillator screen.

counts_per_joule - Number of counts recorded per each joule of energy received by the detector. The number of incident photons can then be calculated by:

$$\text{number of photons} = \frac{\text{source energy} \times \text{data counts}}{\text{counts per joule}}$$

basis_vectors - A matrix with the basis vectors of the detector data. For more details see ??.

corner_position - The x, y and z coordinates of the corner of the first data element. For more details see ??.

geometry_1 - Position and orientation of the center of mass of the detector. This should only be specified for non pixel detectors. For pixel detectors use **basis_vectors** and **corner_position**.

6.1.2 ROI

Class describing the region of interest (ROI) of the image actually collected, if smaller than the full CCD.

Table 24: roi class members

Member	Type	Example
name	string	"APS"
x1	integer	256
y1	integer	256
x2	integer	1792
y2	integer	1792

x1 - Left pixel position.

y1 - Top pixel position.

x2 - Right pixel position.

y2 - Bottom pixel position.

6.1.3 Objective

Class describing the microscope objective lenses used.

Table 25: objective class members

Member	Type	Example
manufacturer	string	"Zeiss"
model	string	"Axioplan"
magnification	float	5
na	float	0.8

manufacturer - Lens manufacturer.

model - Lens model.

magnification - Lens specified magnification.

na - The numerical aperture (N.A.) is a measure of the light-gathering characteristics of the lens.

6.1.4 Scintillator

Class describing the visible light scintillator coupled to the CCD camera objective lens.

Table 26: scintillator class members

Member	Type	Example
manufacturer	string	"Crytur"
serial_number	string	"12"
name	string	"Yag polished"
type	string	"Yag on Yag"
scintillating_thickness	float	5e-6
substrate_thickness	float	1e-4

manufacturer - Scintillator Manufacturer.

serial_number - Scintillator serial number.

name - Scintillator name.

scintillating_thickness - Scintillator thickness.

substrate_thickness - Scintillator substrate thickness.

6.1.5 Setup

Class storing the positions of the stack of stages located under the sample before the tomographic data collection start. The stack defined in Table 27 is used at the APS and the SLS and consists of an x-y-z stack and a pitch-roll (rotation_x and rotation_y) located under the rotary stage, plus an x-z (xx and zz) located above the rotary stage.

Table 27: Sample stack class members

Member	Type	Example
x_coordinate	float	6.6E-3
y_coordinate	float	4.3E-3
z_coordinate	float	5.5E-3
xx_coordinate	float	-8.1E-3
zz_coordinate	float	1.6E-3
rotation_x	float	0.00
rotation_z	float	0.00

[x_coordinate](#), [y_coordinate](#), [z_coordinate](#) - Position of the x-y-z stages located under the rotary stage at the beginning of the scan.

[xx_coordinate](#), [zz_coordinate](#) - Position of the x and z stages located above the rotary stage.

[rotation_x](#), [rotation_z](#) - Position of pitch and roll stages located under the rotary stage

6.1.6 Rotation Setup

This class stores the rotary stage parameters.

Table 28: Rotation Setup class members

Member	Type	Example
start_angle	float	0.000
end_angle	float	180.000
angular_step	float	0.125
angular_speed	float	0.2

start_angle - Rotary stage position at the beginning of the scan.

end_angle - Rotary stage position at the end of the scan.

angular_step - Rotary stage step size (used if data are collected in a stop-go mode).

angular_speed - Rotary stage speed.

6.1.7 Acquisition

A tomographic data set consists of a series of projections, dark and white field images. The dark field and white fields can be collected at any time before, after or during the projection data collection. The acquisition class stores scan parameter values associated with the tomographic data collection.

Table 29: acquisition class members

Member	Type	Example
type	string	"stop and go"
start_date	string ISO 8601	"2011 07 15T15 10Z"
end_date	string ISO 8601	"2011 07 15T25 10Z"
number_of_projections	integer	1441
dark_setup	Dark Setup Class	
white_setup	White Setup Class	
rotation_setup	Rotation Setup Class	

type - Tomographic data collection type: stop and go, fly scan etc.

start_date - Tomographic data collection start.

end_date - Tomographic data collection end.

number_of_projections - Number of tomographic projections.

dark_setup - Dark field data collection setup.

white_setup - White field data collection setup.

rotation_setup - Rotation stage setup.

6.1.8 Dark Setup

This class stores the parameters used to collect the dark field images

Table 30: Dark Setup class members

Member	Type	Example
frequency	int	0
period	int	0
number_pre	int	1
number_post	int	1

frequency - The frequency of dark image collection during rotation. Specified as the number of regular projections to take prior to taking a number of dark images given by period. For example, a value of 10 means to take 10 projections, and then one or more dark images.

period - The number of dark images to collect during rotation at intervals specified by frequency.

number_pre - Number of dark images collected pre-rotation.

number_post - Number of dark images collected post-rotation.

6.1.9 White Setup

This class stores the parameters used to collect the white field images.

Table 31: White Setup class members

Member	Type	Example
frequency	int	0
period	int	0
number_pre	int	1
number_post	int	1
in_out_axis	string	"X"
in	float	0
out	float	3e-3

frequency - The frequency of dark image collection during rotation. Specified as the number of regular projections to take prior to taking a number of dark images given by period. For example, a value of 10 means to take 10 projections, and then one or more dark images.

period - The number of dark images to collect during rotation at intervals specified by frequency.

number_pre - Number of dark images collected pre-rotation.

number_post - Number of dark images collected post-rotation.

in_out_axis - Indicates which axis is used to move the sample out of the field of view.

in - Position of the in_out_axis when the sample is in the data collection position.

out - Position of the in_out_axis when the sample is outside of the field of view.

6.1.10 Interferometer

This class stores the interferometer parameters.

Table 32: Interferometer class members

Member	Type	Example
start_angle	float	0.000
grid.start	float	0.000
grid.end	float	2.4e-6
grid.position_for_scan	float	1.3e-6
number_of_grid_steps	integer	8

start_angle - Interferometer start angle.

grid.start - Interferometer grid start angle.

grid.end - Interferometer grid end angle.

grid.position_for_scan - Interferometer grid position for scan.

number_of_grid_steps - Number of grid steps.

6.2 APS 2-BM Process descriptions

For the APS 2-BM tomography system we define the following process descriptions:

6.2.1 Sinogram

The sinogram class contains all information and parameters required to generate sinograms from projection data.

Table 33: sinogram class members

Member	Type	Example
name	string	
version	string	1.0
input_data	string	"/exchange_1"
input_data_axes	string	"theta:y:x"
output_data	string	"/exchange_2"
output_data_axes	string	"y:theta:x"

name - Algorithm name.

version - Algorithm version.

input_data - Path to the input data.

input_data_axes - Input 3D array axes order.

output_data - Path to the output data.

output_data_axes - Output 3D array axes order.

6.2.2 Ring Removal

The ring removal class contains information required to run a ring removal processing step.

Table 34: Ring removal class members

Member	Type	Example
name	string	
version	string	1.0
input_data	string	"/exchange_2"
output_data	string	"/exchange_2"
coefficient	float	1.0

name - Algorithm name.

version - Algorithm version.

input_data - Path to the input data.

output_data - Path to the output data.

coefficient -

6.2.3 Reconstruction

The Reconstruction class contains all information and parameters required to run a tomography reconstruction using the APS cluster.

Table 35: Reconstruction class members.

Member	Type	Example
input_data	string	"/exchange.2"
output_data	string	"/exchange.3"
reconstruction_time	float	37.5
reconstruction_slice_start	int	1000
reconstruction_slice_end	int	1030
rotation_center	float	1048.50
algorithm	Algorithm class	

input_data - Path to the input data.

output_data - Path to the output data. This parameter if supported by the reconstruction code will be used to save the reconstructed data in the HDF5 file.

reconstruction_time - Total time (s) to reconstruct the full data set.

reconstruction_slice_start - First reconstruction slice.

reconstruction_slice_end - Last reconstruction slice.

rotation_center - Center of rotation in pixels.

algorithm - Algorithm class describing reconstruction algorithm parameters.

6.2.4 Algorithm

The Algorithm class contains information required to run a tomography reconstruction using the APS cluster.

Table 36: Algorithm class members

Member	Type	Example
name	string	SART
version	string	1.0
implementation	string	GPU
number_of_nodes	integer	16
type	string	Iterative
iterative_stop_condition	string	iteration_max
iterative_iteration_max	integer	200
iterative_projection_threshold	float	
iterative_difference_threshold_percent	float	
iterative_difference_threshold_value	float	
iterative_regularization_type	string	total_variation
iterative_regularization_parameter	float	
iterative_step_size	float	0.3
iterative_sampling_step_size	float	0.2
analytic_filter	string	"Parzen"
analytic_padding	float	0.50
analytic_processed_periods	float	1
analytic_processed_number_of_steps	integer	7

name - Reconstruction method name: SART, EM, FBP, GridRec.

version - Algorithm version.

implementation - CPU or GPU.

number_of_nodes - Number of nodes used. This parameter is set when the reconstruction is parallelized and run on a cluster.

type - Tomography reconstruction method: analytic or iterative.

iterative_stop_condition - iteration_max, projection_threshold, difference_threshold_percent, difference_threshold_value.

iterative_iteration_max - Maximum number of iterations.

iterative_projection_threshold - The threshold of projection difference to stop the iterations as p in $|y - Ax_n| < p$.

iterative_difference_threshold_percent - The threshold of reconstruction difference to stop the iterations as p in $|x_{n+1}|/|x_n| < p$.

iterative_difference_threshold_value - The threshold of reconstruction difference to stop the iterations as p in $|x_{n+1}| - |x_n| < p$.

iterative_regularization_type - total_variation, none.

iterative_regularization_parameter - lambda/alpha value in $(y - A_x)^2 + \alpha * L_1(x)$.

iterative_step_size - Step size between iterations in iterative methods as δ_t in $x_{n+1} = x_n + \delta_t * f(x_n)$.

iterative_sampling_step_size - Step size used for forward projection calculation in iterative methods.

analytic_filter - Filter type.

analytic_padding -

analytic_processed_periods - number of processed periods of the collected phase stepping curve (differential phase contrast - grating).

analytic_processed_number_of_steps - total number of processed phase steps (differential phase contrast - grating).

6.2.5 Gridftp

The `gridftp` class contains all information and parameters required to transfer data using the APS `gridftp` server.

Table 37: `gridftp` class members

Member	Type	Example
<code>name</code>	string	
<code>version</code>	string	1.0
<code>input_data</code>	string	"/source file location"
<code>output_data</code>	string	"/remote user cluster location"
<code>credentials</code>	string	"anonymous"

`name` - Algorithm name.

`version` - Algorithm version.

`input_data` - Path to the input data.

`output_data` - Path to the output data.

`credentials` - Account/credentials used for the data transfer.

6.2.6 Export

The export class contains all information and parameters required to extract and export data from a Data Exchange file.

Table 38: export class members

Member	Type	Example
name	string	
version	string	1.0
input_data	string	"/exchange_3"
output_data	string	"/user_folder"
output_data_format	string	"TIFF"
output_data_scaling_max	float	0.005
output_data_scaling_min	float	-0.00088
output_data_prefix	string	"cells sample 1"

name - Algorithm name.

version - Algorithm version.

input_data - Path to the input data.

output_data - Path to the output data.

output_data_format -

output_data_scaling_max -

output_data_scaling_min -

output_data_prefix -

6.3 Creating a minimal Data Exchange file for tomography

Include code here

The resulting file should be equivalent to the one in Fig. 4.

6.4 Creating a typical Data Exchange file for tomography

Include code here

The resulting file should be equivalent to the one in Fig. 5.

7 Data Exchange for X-ray Photon Correlation

In x-ray photon correlation the arrays representing the most basic version of the data include

Table 39: exchange class members for x-ray photon correlation

Member	Type	Example
title	string	"raw photon correlation data"
data	array	see ?? for attributes

title - This is the data title.

data - An x-ray photon correlation data set consists of

7.1 Instrument specific for x-ray photon correlation

7.1.1 Detector

This class holds information about the detector used during the experiment. If more than one detector are used they will be all listed as `detector_N`.

manufacturer - The detector manufacturer.

model - The detector model.

serial_number - The detector serial number .

bit_depth - The detector bit depth.

x_pixel_size, y_pixel_size - Physical detector pixel size (m).

x_dimension, y_dimension - The detector horiz./vertical dimension.

x_binning, y_binning - If the data are collected binning the detector **x_binning** and **y_binning** store the binning factor.

operating_temperature - The detector operating temperature (K).

exposure_time - The detector exposure time (s).

exposure_period - Time from the beginning of an exposure to the beginning of the next exposure (s).

frame_rate - The detector frame rate (fps). This parameter is set for fly scan

Table 40: X-ray photon correlation detector class members

Member	Type	Example
manufacturer	string	"Cooke Corporation"
model	string	"pco dimax"
serial_number	string	"1234XW2"
bit_depth	integer	12
x_pixel_size	float	6.7e-6
y_pixel_size	float	6.7e-6
x_dimension	integer	2048
y_dimension	integer	2048
x_binning	integer	1
y_binning	integer	1
operating_temperature	float	270
exposure_time	float	1.7e-3
exposure_period	float	10.0
frame_rate	integer	2
distance	float	5.7e-3
data_flat	3D array	variable (see Tab. 4 for attrib.)
roi	roi class	
efficiency	float	99.95
adu_per_photon	float	5.0
gain	float	1.0
basis_vectors	float array	length
corner_position	3 floats	length
blemish_mask	2D array of integers	variable (see Tab. 4 for attrib.)
kinetics	kinetics class	
geometry	string	"TRANSMISSION"

distance - The detector distance from the sample.

data_flat - The dark field and white fields must have the same dimensions as the collected images. Data_flat attributes, if used, are defined in Table 4.

roi - The detector selected Region Of Interest (ROI).

efficiency - The efficiency of the detector.

adu_per_photon - The ADU per photon.

gain - Detector gain setting.

basis_vectors - A matrix with the basis vectors of the detector data. For more details see ??.

corner_position - The x, y and z coordinates of the corner of the first data element. For more details see ??.

blemish_mask - Blemish mask labeling dead pixels in the detector. It's a 2D array of the same dimensions as the full detector, with 0 labeling bad pixels and 1 labeling good pixels.

kinetics - Kinetics detector properties.

geometry - TRANSMISSION or REFLECTION.

7.1.2 Kinetics

Class describing the kinetics camera mode properties. This applies only to kinetics mode cameras.

Table 41: kinetics class members

Member	Type	Example
name	string	"APS"
window_size	integer	256
top	integer	1024
first_usable_window	integer	1
last_usable_window	integer	4

window_size - Number of rows in each kinetics window.

top - Top pixel.

first_usable_window - The first usable kinetics window.

last_usable_window - The last usable kinetics window.

7.1.3 Setup

This class stores XPCS setup parameters.

Table 42: XPCS Setup class members

Member	Type	Example
beam_center_x	float	0.000
beam_center_y	float	180.000
beam_size_h	float	10.000
beam_size_v	float	18.000
stage_zero_x	float	199.0
stage_zero_z	float	185.0
stage_x	float	199.0
stage_z	float	185.0
xspec	float	0.0
yspec	float	0.0
ccdxspec	float	0.0
ccdyspec	float	0.0
x	float	50
y	float	50
angle	float	50.0

`beam_center_x` - Pixel location of beam center.

`beam_center_y` - Pixel location of beam center.

`beam_center_h` - Horizontal beam size on detector.

`beam_center_v` - Vertical beam size on detector.

`stage_zero_x` - Initial stage location.

`stage_zero_y` - Initial stage location.

`stage_x` - Stage location.

`stage_y` - Stage location.

`xspec` - ...

`yspec` - ...

`ccdxspec` - ...

`ccdyspec` - ...

`x` - ...

`y` - ...

angle - ...

7.2 APS Sector 8 Process descriptions

For the APS Sector 8 x-ray photon correlation system we define the following process descriptions:

7.2.1 XPCS

The XPCS class contains all information and parameters required to run a using the APS cluster.

Table 43: XPCS class members.

Member	Type	Example
input_file_local	string	"InputFile.imm"
output_file_local	string	"OutputFile.imm"
input_file_remote	string	"InputFile.imm"
output_file_remote	string	"OutputFile.imm"
specfile	string	"Specfile.spec"
specscan_dark_number	integer	
specscan_data_number	integer	
compression	string	"SPARSE"
file_mode	string	"MULTI"
delays_per_level	integer	
lld	float	
sigma	float	
analysis_type	string	"DYNAMIC"
batches	integer	"2"
data_begin	integer	"1"
data_end	integer	"99998"
dark_begin	integer	"99999"
dark_end	integer	"100000"
data_begin_todo	integer	"1"
data_end_todo	integer	"99998"
dark_begin_todo	integer	"99999"
dark_end_todo	integer	"100000"
mask	2D array of integers	variable (see Tab. 4 for attrib.)
dqmap	2D array of floats	variable (see Tab. 4 for attrib.)
sqmap	2D array of floats	variable (see Tab. 4 for attrib.)
dphimap	2D array of floats	variable (see Tab. 4 for attrib.)
sphimap	2D array of floats	variable (see Tab. 4 for attrib.)
dqspan	1D array of floats	variable (see Tab. 4 for attrib.)
dphispan	1D array of floats	variable (see Tab. 4 for attrib.)
sqspan	1D array of floats	variable (see Tab. 4 for attrib.)
sphispan	1D array of floats	variable (see Tab. 4 for attrib.)
splist	1D array of floats	variable (see Tab. 4 for attrib.)
dqlist	1D array of floats	variable (see Tab. 4 for attrib.)
sphillist	1D array of floats	variable (see Tab. 4 for attrib.)
dphillist	1D array of floats	variable (see Tab. 4 for attrib.)
normalization_method	string	"INCIDENT"
blemish_enabled	string	"TRUE"
flatfield_enabled	string	"TRUE"

output_file_local - Full path for output data file.
input_file_local - Path to the input data file.
output_file_local - Full path for output data file.
specfile - Full path to spec file.
specscan_dark_number - ...
specscan_data_number - ...
compression - Compression type: either SPARSE or NONSPARSE.
file_mode - MULTI or SINGLE.
delays_per_level - Delays per level.
lld - LLD.
sigma - Sigma.
analysis_type - STATIC or DYNAMIC
batches - Number of batches.
data_begin - Index of first data frame in .imm file.
data_end - Index of last data frame in .imm file.
dark_begin - Index of first dark data frame in .imm file.
dark_end - Index of last dark data frame in .imm file.
data_begin_todo - Index of first data frame in .imm file to analyze.
data_end_todo - Index of last data frame in .imm file to analyze.
dark_begin_todo - Index of first dark data frame in .imm file to analyze.
dark_end_todo - Index of last dark data frame in .imm file to analyze.
mask - Mask used to exclude image regions during analysis. It's a 2D array of the same dimensions as the collected data, with 0 labeling excluded pixels and 1 labeling included pixels.
dqmap - dqmap. 2D array of the same dimensions as the collected data.

sqmap - sqmap. 2D array of the same dimensions as the collected data.

dphimap - dphimap. 2D array of the same dimensions as the collected data.

sphimap - sphimap. 2D array of the same dimensions as the collected data.

dqspan - dqspan. 1D array.

dphispan - dphispan. 1D array.

sqspan - sqspan. 1D array.

sphispan - sphispan. 1D array.

sqlist - sqlist. 1D array.

dqlist - dqlist. 1D array.

sphilist - sphilist. 1D array.

dphilist - dphilist. 1D array.

normalization_method - Normalization method: INCIDENT or TRANSMITTED or BOTH or NONE.

blemish_enabled - Use detector blemish data.

flatfield_enabled - Use detector flat field data.

8 Data Exchange for X-ray Fluorescence

In x-ray fluorescence the arrays representing the most basic version of the data include

Table 44: exchange class members for x-ray fluorescence

Member	Type	Example
title	string	"raw fluorescence data"
data	array	see ?? for attributes

title - This is the data title.

data - An x-ray fluorescence data set consists of

8.1 Instrument specific for x-ray fluorescence

A Appendix

A.1 Default units for Data Exchange entries

The default units for Data Exchange entries follow the CXI entries definition, i.e. are SI based units (see table 45) unless the "units" attribute is specified. Data Exchange prefers not to use "units" and use the default SI based units whenever possible.

Table 45: SI (and common derived) base units for different quantities

Quantity	Units	Abbreviation
length	meter	m
mass	kilogram	kg
time	second	s
electric current	ampere	A
temperature	kelvin	K
amount of substance	mole	mol
luminous intensity	candela	cd
frequency	hertz	Hz
force	newton	N
pressure	pascal	Pa
energy	joule	J
power	watt	W
electric potential	volt	V
capacitance	farad	F
electric resistance	ohm	Ω
absorbed dose	gray	Gy
area	square meter	m ²
volume	cubic meter	m ³

A.1.1 Angles

Angles are always defined in degrees *not* in radians.

A.1.2 Dates

Dates are always specified according to the [ISO 8601](#). This means for example "1996-07-31T21:15:22+0600". Note the "T" separating the data

from the time and the “+0600” timezone specification.

A.2 Geometry

A.2.1 Coordinate System

The Data Exchange uses the same CXI coordinate system. This is a right handed system with the z axis parallel to the X-ray beam, with the positive z direction pointing away from the light source, in the downstream direction. The y axis is vertical with the positive direction pointing up, while the x axis is horizontal completing the right handed system (see Fig. 9). The origin of the coordinate system is defined by the point where the X-ray beam meets the sample.

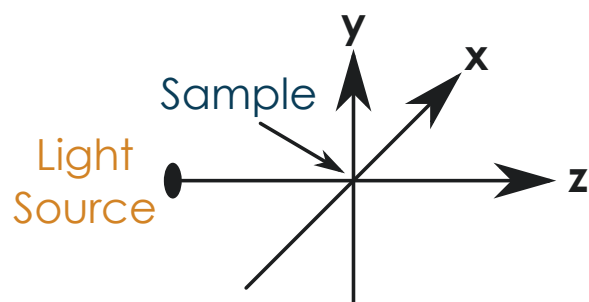


Figure 9: The coordinate system used by CXI. The intersection of the X-ray beam with the sample define the origin of the system. The z axis is parallel to the beam and points downstream.

A.2.2 The local coordinate system of objects

For many detectors their location and orientation is crucial to interpret results. Translations and rotations are used to define the absolute position of each object. But to be able to apply these transformations we need to know what is the origin of the local coordinate system of each object. Unless otherwise specified the origin should be assumed to be the geometrical center of the object in question. The default orientation of the object should have the longest axis of the object aligned with the x axis, the second longest with the y axis and the shortest with the z axis.